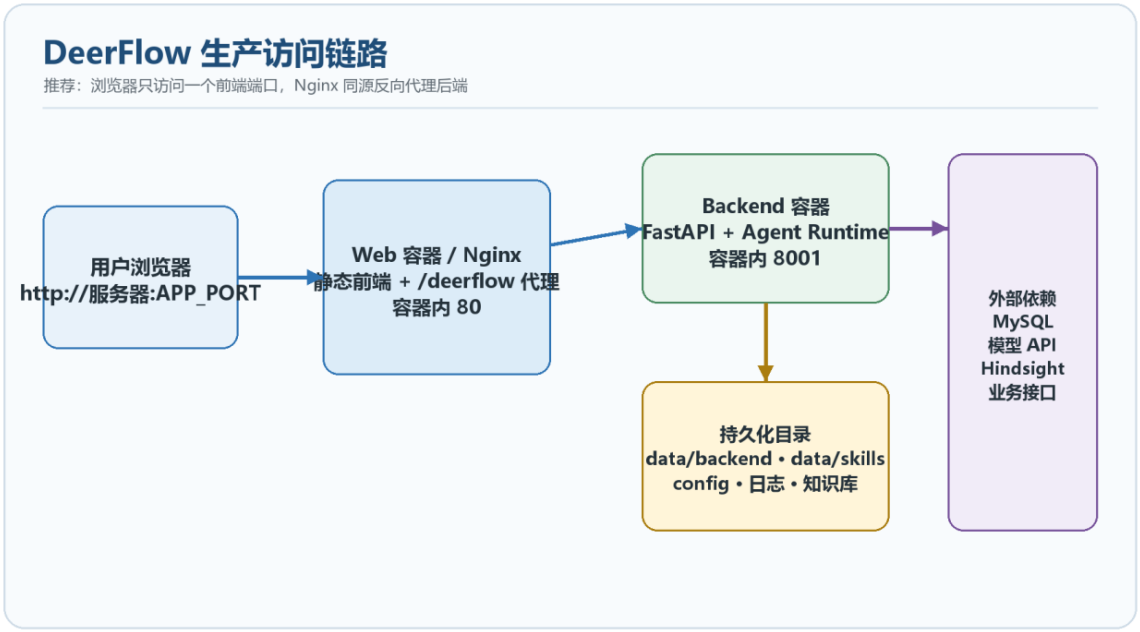
DeerFlow

系统运维手册

内网离线部署版 • Docker / 1Panel / ARM64



系统生产访问链路总览

> **适用对象** 首次接手本系统的 Linux/1Panel 运维人员、DeerFlow 应用管理员、模型与数据管理员。

文档版本	适用版本	更新日期
V1.0	当前离线部署包与本项目定制版	2026-08-08

密级：内部运维资料。本文所有密钥、密码、IP 和数据库账号均使用占位符。

文档导航

本文按“先理解、再配置、后维护、能恢复”的顺序编排。Word 左侧导航窗格可按标题跳转；如目录页码未自动刷新，请在 Word 中按 Ctrl+A 后按 F9。

版本与变更记录 8

阅读约定 8

1. 系统全貌与运维边界 9

1.1 服务与端口 9

1.2 需要纳入监控和备份的对象 10

1.3 本手册不替代的制度 10

2. 账号、角色与登录密钥 11

2.1 建议准备的账号 12

2.2 AUTH_JWT_SECRET 的作用 12

2.3 用户名登录密码与 JWT 密钥的区别 13

2.4 管理员与 lqq 的判定 13

3. 部署目录、配置分层与变更生效 14

3.1 标准部署目录 15

3.2 变更生效速查 15

3.3 修改前的通用流程 15

4. Docker / 1Panel 日常启停与日志 17

4.1 一键安装、重启、停止 17

4.2 只重启后端或前端 17

4.3 打印实时日志 18
4.4 一键诊断 18
4.5 健康检查 18
5. 后端模型配置与 max_tokens 19
5.1 新增一个 OpenAI 兼容模型 19
5.2 字段说明 20
5.3 如何添加或调整 max_tokens 21
5.4 模型变更后的验证 21
5.5 模型故障快速定位 21
6. 数据库：SQLite 与外部 MySQL 23
6.1 默认 SQLite 23
6.2 切换到外部 MySQL 23
6.3 MySQL 前置条件 24
6.4 切换与回退步骤 24
6.5 数据库常见报错 25
7. 前端端口、接口地址与运行时配置 26
7.1 修改前端访问端口 26
7.2 推荐的同源后端配置 26
7.3 后端独立地址的配置 26
7.4 常用运行时变量 27
7.5 修改后的生效 27
7.6 前端常见现象 28

8. 技能：目录、上传、发布与维护 29
8.1 标准技能目录 29
8.2 推荐：通过前端上传或创建 30
8.3 手工离线导入技能 30
8.4 public 与广场发布的区别 30
8.5 技能地址管理 31
8.6 技能发布到广场 31
8.7 技能上传安全检查 31
9. 智能体：目录、模型、发布与恢复 32
9.1 标准目录与职责 32
9.2 创建与维护建议 32
9.3 发布到广场 32
9.4 直接发布与私有广场 33
9.5 智能体模型失效 33
10. 发布广场与私有广场管理 34
10.1 管理员配置广场 34
10.2 发布规则 34
10.3 看不到发布按钮 34
11. 菜单名称、排序、隐藏与路由安全 35
11.1 修改菜单名称 35
11.2 隐藏菜单 36
11.3 需要真正禁用功能时 36

11.4 菜单异常恢复 36
12. 主题色、布局和品牌配置 37
12.1 内置主题 37
12.2 管理员设置系统默认 37
12.3 自定义主题色的验收 37
12.4 强制紧凑侧边栏 38
12.5 iframe 嵌入主题 38
13. 敏感词管理与显示层脱敏 39
13.1 登录哪个用户 39
13.2 操作步骤 39
13.3 设计与风险 39
13.4 页面看不到时 40
14. 备份、恢复、升级与回滚 41
14.1 备份范围 41
14.2 文件备份示例 42
14.3 MySQL 备份示例 42
14.4 恢复原则 42
14.5 离线升级 42
14.6 备份校验与恢复演练 43
15. 监控、巡检与日志留存 44
15.1 每日巡检 44
15.2 每周巡检 44

15.3 建议监控指标 44
15.4 日志采集包 45
16. 常见报错与解决办法 46
16.1 容器和主机 46
16.2 前端和 Nginx 46
16.3 登录和权限 47
16.4 模型和流式对话 47
16.5 数据库和资产 47
17. 技能或智能体丢失专项排查 49
17.1 第一步：确认当前服务和部署目录 49
17.2 第二步：确认登录身份 50
17.3 第三步：核对宿主机与容器内文件 50
17.4 第四步：确认数据库没有切换 50
17.5 第五步：检查过滤、广场和权限 50
17.6 第六步：日志和权限 51
17.7 恢复决策 51
18. 安全加固与密钥管理 52
18.1 密钥清单 52
18.2 最低安全要求 52
18.3 日志脱敏 52
18.4 离线介质 53
19. 标准运维流程与交接清单 54

19.1 上线验收 54
19.2 变更单最少信息 54
19.3 离职/交接 54
19.4 运维责任矩阵 55
19.5 交接会必须确认 55
附录 A：命令速查 56
A.1 服务 56
A.2 日志与健康 56
A.3 主机资源 56
A.4 文件与挂载 56
附录 B：安全配置模板 57
B.1 部署 .env 57
B.2 backend.env 57
B.3 runtime-config.js 57
B.4 模型条目 58
附录 C：问题升级模板 59
C.1 现场信息保全 59
附录 D：核心事实速记 60

> **先看这四章** 首次接手请先读第 1、2、3、4 章；发生资产丢失或生产故障时直接跳到第 16、17 章。

版本与变更记录

版本	日期	变更说明	维护人
V1.0	2026-08-08	首次发布，覆盖离线部署、配置、资产发布、备份恢复与排错	DeerFlow 运维团队

阅读约定

- “部署目录”指解压离线包后，包含 docker-compose.yml、config、data、restart.sh 等内容的目录。
- 命令默认以 Linux shell 执行；1Panel 的容器终端或 Xshell 均可。
- 涉及生产数据的命令，先确认当前路径，再做备份；不要机械复制带删除含义的命令。
- 本文以 APP_PORT=8090 为例，实际端口以部署目录 .env 为准。

1. 系统全貌与运维边界

DeerFlow 是由 React 前端、Nginx、FastAPI/LangGraph 后端、持久化文件目录和业务数据库共同组成的 AI 智能体平台。前端和后端在容器内使用不同端口，但生产访问通常只暴露一个 Web 端口：浏览器访问 Nginx，Nginx 再把 /deerflow 请求转发到后端。

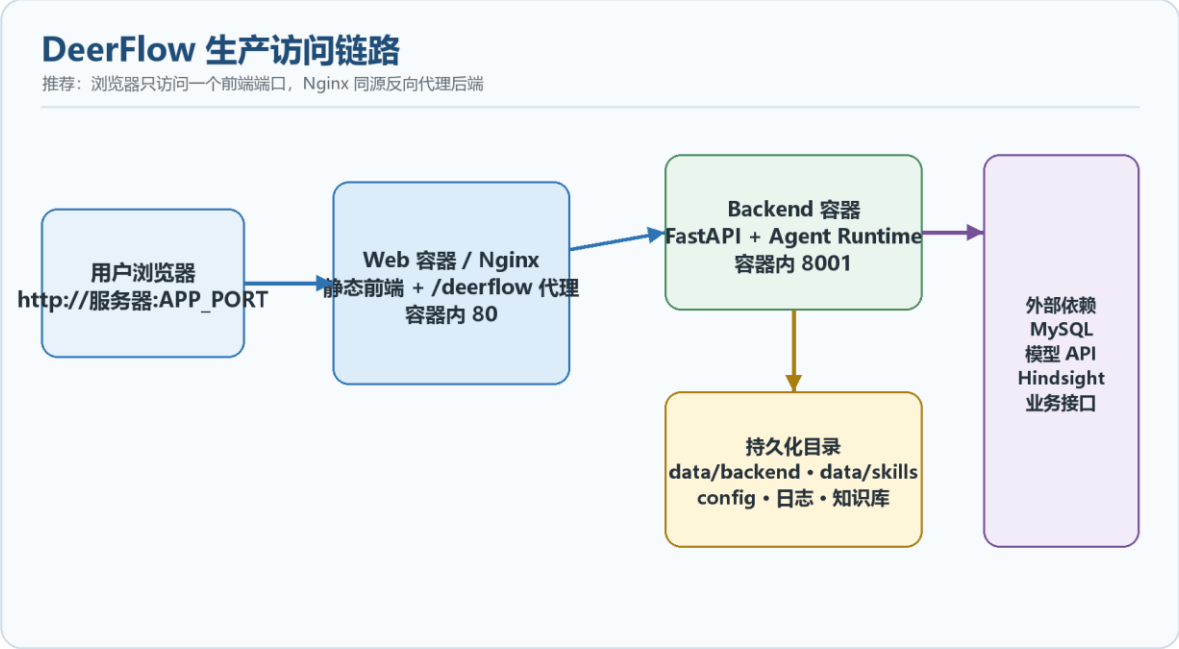


图 1-1 生产访问与数据流

1.1 服务与端口

组件	容器内端口	宿主机端口	主要职责	运维结论
Web / Nginx	80	APP_PORT（默认 8090）	前端静态文件、SPA 路由、反向代理	公网或内网只需开放这一端口
Backend	8001	默认不直接映射	登录、智能体、技能、模型调用、业务 API	通过 /deerflow 由 Nginx 访问
MySQL（可选）	由数据库决定	通常仅内网	业务表、资产元数据、用户与全局配置	容器要能访问数据库地址

> **关键结论** 前端与后端不是同一个容器端口。对用户而言可以是同一个访问端口，因为 Nginx 做了同源代理。

1.2 需要纳入监控和备份的对象

- 部署配置：.env、config/backend.env、config/config.yaml、config/extensions_config.json、config/runtime-config.js。
- 持久化文件：data/backend、data/skills、data/knowledge、data/logs。
- 数据库：默认 SQLite 文件；切换外部 MySQL 后还要做 MySQL 备份。
- 离线镜像与版本：保留镜像 tar 包、SHA256、部署日期和变更记录。
- 外部依赖：模型 API、MySQL、Hindsight、MCP、企业业务接口的地址和账号。

1.3 本手册不替代的制度

生产变更审批、账号实名、密码轮换、数据库备份保留期和等保要求仍以本单位制度为准。本文提供技术步骤，不构成放宽安全控制的依据。

2. 账号、角色与登录密钥

运维中最容易混淆的是“服务器 root”和“DeerFlow admin”。前者能管理容器和文件，后者才能在应用页面中配置菜单、主题、用户和发布广场。敏感词页面又有独立的 lqq 账号入口条件。



图 2-1 四类身份与职责

2.1 建议准备的账号

账号类型	至少数量	用途	注意事项
Linux / 1Panel 管理员	2	部署、日志、备份、恢复	不共用个人账号；禁止把 root 密码写进文档
DeerFlow admin	2	全局设置、白名单、菜单、主题、广场	system_role 必须是 admin；保留一名应急管理员
lqq	1	前端敏感词管理	显示用户名或邮箱 @ 前缀必须严格为 lqq
普通测试用户	1	验证普通权限、资产发布和页面可见性	不能只用管理员做验收
模型/数据库服务账号	按环境	调用外部服务	最小权限、单独轮换、不得与人员账号共用

2.2 AUTH_JWT_SECRET 的作用

AUTH_JWT_SECRET 是后端签发和校验登录令牌的服务端密钥，不是用户登录密码。它必须在多进程、多容器和每次重启之间保持一致。若更换该值，所有现有登录会话会失效，需要重新登录。

ENV

```
# config/backend.env
AUTH_JWT_SECRET=<至少 32 字节的高强度随机值>
```

在可联网或可信管理机生成随机值，再通过受控介质导入内网。Linux 可使用：

Shell

```
openssl rand -hex 32
```

> **禁止事项** 不要把密钥提交到 Git，不要写入 runtime-config.js，不要保留在 config.yaml 的注释中。浏览器可下载的前端配置中永远不能放服务端密钥。

2.3 用户名登录密码与 JWT 密钥的区别

项目	保存位置	谁使用	变更影响
AUTH_JWT_SECRET	config/backend.env	后端服务	全部登录会话失效
用户名登录共享密码	config/config.yaml 的 auth_login.username_login	登录入口校验	影响使用该入口的新登录
账号 login_password	应用用户/白名单管理	指定账号	只影响该账号
VITE_ROUTE_LOGIN_PASSWORD	runtime-config.js 或构建环境	前端特定路由辅助	不是后端安全边界

2.4 管理员与 lqq 的判定

- 应用管理员由 system_role=admin 判定；内置 admin 用户通常会被提升为管理员，但仍应在用户管理页核验。
- 敏感词写接口允许管理员或 lqq，但当前前端敏感词页面的入口和路由会检查显示用户名是否为 lqq。因此正常页面维护建议直接使用 lqq 登录。
- 邮箱账号以 @ 前缀作为显示用户名时，应确保前缀为 lqq；不要使用类似 lqq01 代替。

3. 部署目录、配置分层与变更生效

同一项配置可能存在源码默认值、镜像内默认值和运行时覆盖值。生产运维只改部署目录中的运行时文件，并记录变更；不要进入容器直接改文件，因为重建容器后会丢失。



图 3-1 配置分层和生效方式

3.1 标准部署目录

目录结构

```
deerflow-deploy/
├── docker-compose.yml
├── .env
├── config/
│   ├── backend.env
│   ├── config.yaml
│   ├── extensions_config.json
│   └── runtime-config.js
├── data/
│   ├── backend/
│   ├── skills/
│   ├── knowledge/
│   └── logs/
├── install.sh
├── restart.sh
├── stop.sh
├── logs.sh
└── diagnose.sh
```

3.2 变更生效速查

修改对象	典型内容	最小生效动作	验证
.env	APP_PORT、镜像标签	重建受影响容器	端口监听、浏览器访问
backend.env	JWT、模型 Key、MySQL URL	强制重建 Backend	后端日志、/health、实际功能
config.yaml	模型、数据库模式、登录配置	重启 Backend	启动日志、模型列表、登录
runtime-config.js	前端 API/业务接口、Logo	建议重启 Web；浏览器 Ctrl+F5	浏览器 Network 与页面
管理页面	菜单、主题、敏感词、广场	保存后通常即时生效；必要时刷新	管理员和普通用户分别验证

3.3 修改前的通用流程

1. 确认当前部署目录和 Compose 项目，不要在旧目录操作。
2. 复制要修改的配置文件并加时间戳备份。
3. 记录修改人、工单、原值、目标值和回退方法。

4. 只重启受影响的服务；变更后执行健康检查和实际业务验证。
5. 观察日志至少 5 分钟，再结束变更窗口。

4. Docker / 1Panel 日常启停与日志

4.1 一键安装、重启、停止

进入部署目录后执行以下脚本。生产环境建议保留 sudo，避免数据目录因不同用户写入而出现权限不一致。

Shell

```
cd /opt/deerflow/deerflow-deploy

# 首次安装 / 加载镜像并启动
sudo bash install.sh

# 一键重启全部服务
sudo bash restart.sh

# 一键停止全部服务
sudo bash stop.sh
```

4.2 只重启后端或前端

Shell

```
# 日常重启后端，或仅 config.yaml / 挂载文件变更
sudo docker compose restart backend

# backend.env 发生变化时，必须重建容器以重新加载环境变量
sudo docker compose up -d --force-recreate backend

# 只重启 Web（前端运行时配置或 Nginx 变更）
sudo docker compose restart web

# 查看容器状态
sudo docker compose ps
```

> **环境变量不会被普通 restart 重新读取** restart.sh 和 docker compose restart 适合日常重启；修改 backend.env、APP_PORT 或其他 Compose 环境变量后，应使用 docker compose up -d --force-recreate <服务>。

> **1Panel 操作** 在 1Panel 的容器/编排页面执行“重启”也可以，但要确认选中的是当前部署目录对应的 Compose 项目。脚本方式更便于保留统一操作记录。

4.3 打印实时日志

Shell

```
# 全部服务实时日志（Ctrl+C 退出，不会停止服务）
sudo bash logs.sh

# 只看后端
sudo bash logs.sh backend

# 只看前端 / Nginx
sudo bash logs.sh web

# Compose 原生命令：最近 500 行并持续跟踪
sudo docker compose logs --tail=500 -f backend
sudo docker compose logs --tail=500 -f web
```

4.4 一键诊断

Shell

```
sudo env TAIL=1000 bash diagnose.sh
```

诊断结果应至少包含架构、磁盘、内存、容器状态、端口监听、健康检查和最近错误日志。发送给研发前先检查输出中是否包含账号、Token、数据库 URL 或 API Key。

4.5 健康检查

Shell

```
# 从服务器本机检查，端口以 .env 的 APP_PORT 为准
curl -fsS http://127.0.0.1:8090/deerflow/health

# 检查前端首页响应头
curl -I http://127.0.0.1:8090/
```

> **不要只看容器 Up** 容器处于 Up 只代表进程存在，不代表模型、数据库和业务功能可用。每次重启后至少完成健康接口、登录和一次对话验证。

5. 后端模型配置与 max_tokens

模型列表由后端 config/config.yaml 管理，密钥放在 config/backend.env。推荐使用 OpenAI 兼容接口时配置 ChatOpenAI 适配器，并逐项核对模型 ID、base_url、超时、重试、输出上限和能力标志。

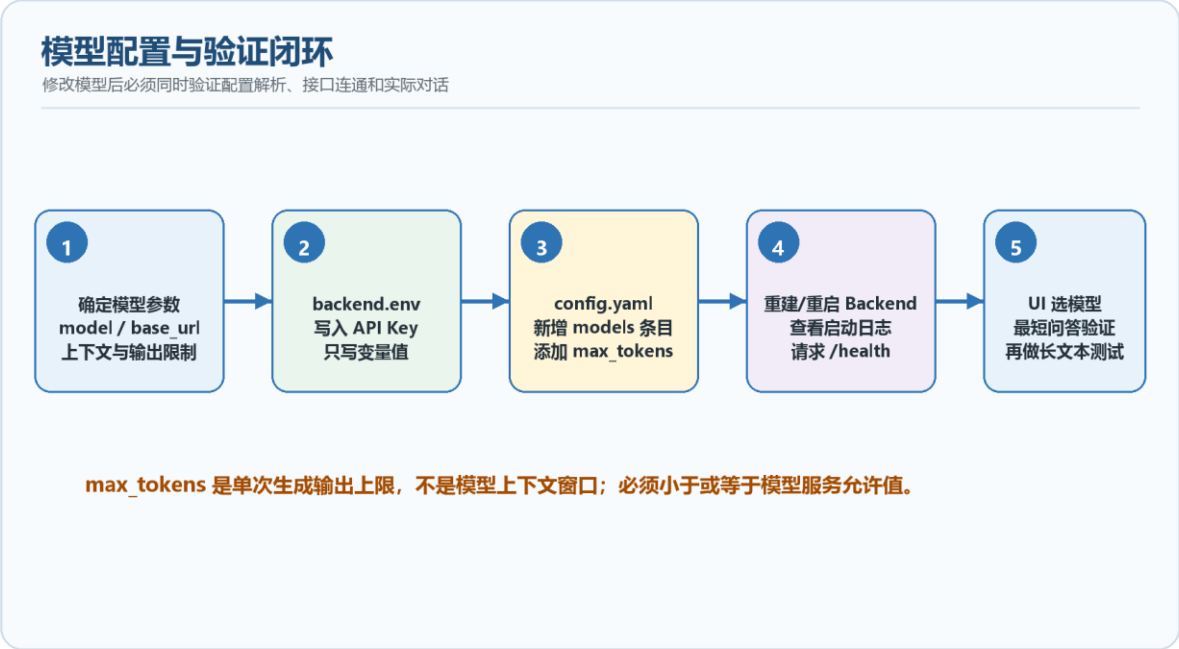


图 5-1 模型配置与验证流程

5.1 新增一个 OpenAI 兼容模型

先在 backend.env 中保存密钥。变量名可自定义，但必须与 config.yaml 中 $ 后的名称完全一致。

ENV

```
# config/backend.env
INTRANET_MODEL_API_KEY=<模型服务分配的密钥>
```

再在 config/config.yaml 的 models 列表中新增条目：

YAML

```yaml
models:
  - name: intranet-model
    display_name: 内网模型
    use: langchain_openai:ChatOpenAI
    model: model-id-from-provider
    api_key: $INTRANET_MODEL_API_KEY
    base_url: http://10.0.0.20:8000/v1
    request_timeout: 600.0
    max_retries: 2
    max_tokens: 8192
    temperature: 0.7
    supports_vision: false
    supports_thinking: false
```

5.2 字段说明

字段	必需	说明	常见错误
name	是	系统内部唯一名称，智能体配置引用它	重名或改名后旧智能体引用失效
display_name	建议	前端显示名称	写成真实密钥或内部地址
use	是	模型适配器，例如 langchain_openai:ChatOpenAI	适配器与接口协议不匹配
model	是	模型服务要求的精确 model id	使用市场名称而非接口 ID，导致 404
api_key	通常	使用 $ENV_NAME 从 backend.env 解析	写成明文、变量名拼错
base_url	通常	OpenAI 兼容接口通常以 /v1 结尾	容器中 127.0.0.1 指向自身
request_timeout	建议	单次请求秒数	长推理模型超时过短
max_retries	建议	网络或限流重试次数	重试过多造成雪崩
max_tokens	建议	单次生成的最大输出 token	超过提供方限制或误认为上下文窗口
supports_vision	按能力	是否允许图像输入	服务不支持却标 true
supports_thinking	按能力	是否支持思考模式	适配器/服务参数不兼容

5.3 如何添加或调整 max_tokens

在目标模型条目中增加 max_tokens: <整数>。它控制模型本次生成的最大输出长度，不等于“输入 + 输出”的总上下文窗口。提供方若只允许 8192 输出，就不能配置 65536。

现象	可能原因	处理
回答被截断	max_tokens 太小、模型自行结束或上游网关限制	逐步增大并检查 finish_reason
400 invalid max_tokens	超过模型上限或字段名不兼容	查模型服务文档/网关日志，改为允许值
长文本仍失败	上下文窗口不足，不是输出上限问题	缩短输入、启用摘要、换长上下文模型
成本或耗时突然上升	上限过大且提示词未约束	降低 max_tokens，并在提示词限定结构和篇幅

5.4 模型变更后的验证

1. 备份 config.yaml 和 backend.env；检查 YAML 缩进，不使用 Tab。
2. 若 backend.env 有变化，执行 sudo docker compose up -d --force-recreate backend；只改 config.yaml 时可 restart backend。
3. 立即跟踪 backend 日志，确认没有 YAML 解析、环境变量缺失或模型初始化错误。
4. 访问 /deerflow/health，登录页面确认新模型出现在可选列表。
5. 完成短问答、长文本、工具调用；视觉模型再验证图片上传。
6. 确认日志中没有 API Key；记录模型版本、base_url、限流和最大上下文。

5.5 模型故障快速定位

HTTP/日志	含义	优先检查
401 / invalid_api_key	模型密钥无效	backend.env、变量名、密钥状态、是否有前后空格
403	密钥无该模型权限或 IP 白名单	模型服务授权与来源 IP

HTTP/日志	含义	优先检查
404 model_not_found	model ID 或 base_url 不对	接口真实 ID、/v1 路径、网关路由
429	限流或余额不足	并发、速率配额、余额、重试策略
timeout	模型慢或网络不通	容器到服务连通性、request_timeout、代理
connection refused	地址/端口错误或服务未启动	不要在容器内使用错误的 127.0.0.1

6. 数据库：SQLite 与外部 MySQL

默认部署可使用 SQLite，便于离线单机运行；已有外部 MySQL 时，可将业务数据库切换到 MySQL。切换数据库不会自动迁移旧 SQLite 中的业务数据，必须先做迁移和验收计划。LangGraph checkpointer 可继续保留 SQLite，两者不是同一配置项。

6.1 默认 SQLite

YAML

```yaml
database:
  backend: sqlite
  sqlite_dir: .deer-flow/data

checkpointer:
  type: sqlite
  connection_string: .deer-flow/data/deerflow.db
```

在容器挂载下，关键数据位于宿主机 data/backend 对应目录。备份时应在写入低峰或停止后端，避免复制过程中数据库仍在变化。

6.2 切换到外部 MySQL

ENV

```env
# config/backend.env
MYSQL_DATABASE_URL=mysql+asyncmy://deerflow_user:<URL 编码后的密码>@10.0.0.30:3306/deerflow?charset=utf8mb4
```

YAML

```yaml
# config/config.yaml
database:
  backend: mysql
  mysql_url: $MYSQL_DATABASE_URL

# 可按现网设计继续使用 SQLite checkpointer
checkpointer:
  type: sqlite
  connection_string: .deer-flow/data/deerflow.db
```

> **容器网络** MySQL 在宿主机或另一台服务器时，地址必须是容器可访问的 IP/域名。容器内的 127.0.0.1 指后端容器自身，不指宿主机。

6.3 MySQL 前置条件

- 数据库已创建，字符集建议 utf8mb4；时区和排序规则按单位标准统一。
- 服务账号至少拥有本应用所需的建表、改表、查询、增删改权限；禁止使用 root 作为长期业务账号。
- 防火墙允许后端容器所在主机访问 3306；MySQL bind-address 和用户 host 范围正确。
- 密码中的 @、:、/、#、% 等字符进行 URL 编码，避免连接串解析错误。
- 首次启动预留数据库迁移时间；观察 Alembic/建表日志，不要在未评估的情况下跳过迁移。

6.4 切换与回退步骤

1. 冻结资产编辑和用户写入，完整备份 SQLite、data 目录与配置。
2. 创建 MySQL 数据库和最小权限账号；从后端容器验证网络。
3. 制定并执行数据迁移。只改 backend=mysql 不会把旧数据自动搬过去。
4. 修改 backend.env 和 config.yaml，执行 docker compose up -d --force-recreate backend，持续看迁移日志。
5. 用 admin、lqq、普通用户分别验收登录、技能、智能体、广场、菜单、敏感词和对话。
6. 确认稳定前保留旧 SQLite 与旧配置；回退时停止写入，避免两边各自产生新数据。

6.5 数据库常见报错

报错	原因	处理
Access denied	账号/密码/host 授权错误	核对 URL 编码、MySQL 用户 host、权限
Unknown database	库未创建或名称错误	创建库并核对大小写
Connection refused / timeout	端口、防火墙、bind-address、路由	从容器执行 TCP 连通测试
Too many connections	连接池或并发超限	先止损降并发，再调连接池和 MySQL 上限
SQLite database is locked	多写者或磁盘/挂载问题	停止并发写，检查权限和磁盘，完整备份后处理
Permission denied	宿主机挂载所有者不正确	核对 data/backend 的 UID/GID 与容器运行用户

7. 前端端口、接口地址与运行时配置

生产前端使用 Nginx，不使用开发端口 5174。对外端口由部署目录 .env 的 APP_PORT 控制；前端访问后端和外部业务接口的地址由 config/runtime-config.js 控制。该文件由浏览器直接读取，不能放任何服务端秘密。

7.1 修改前端访问端口

ENV

```
# 部署目录 .env
APP_PORT=18090
```

Shell

```
sudo docker compose up -d --force-recreate web
sudo docker compose ps
curl -I http://127.0.0.1:18090/
```

1Panel 中若有端口安全组或主机防火墙，还要放行新的 TCP 端口。旧端口如不再使用应关闭。

7.2 推荐的同源后端配置

JavaScript

```
// config/runtime-config.js
window.__DEERFLOW_RUNTIME_CONFIG__ = {
  VITE_BACKEND_BASE_URL: "/deerflow",
  VITE_LANGGRAPH_BASE_URL: "/deerflow/api"
};
```

使用相对路径时，浏览器始终访问当前域名和 APP_PORT，再由 Nginx 转发到 backend:8001。这样最少遇到 CORS、证书和混合内容问题。

7.3 后端独立地址的配置

JavaScript

```
window.__DEERFLOW_RUNTIME_CONFIG__ = {
  VITE_BACKEND_BASE_URL: "https://api.example.internal",
  VITE_LANGGRAPH_BASE_URL: "https://api.example.internal/api"
};
```

> **使用绝对地址时** 后端必须配置允许前端 Origin，HTTPS 页面不能调用 HTTP 接口；证书需被客户端信任。除非确有分离网关需求，优先使用 /deerflow 相对地址。

7.4 常用运行时变量

变量	用途	是否可含秘密	变更验证
VITE_BACKEND_BASE_URL	Gateway 基础地址	否	登录、/health、普通 API
VITE_LANGGRAPH_BASE_URL	LangGraph/流式对话地址	否	新建对话和流式输出
VITE_API_BASE_URL	通用业务 API 根地址	否	对应业务页面
VITE_AIPI_CONSUMER_URL	任务深链/业务查询	否	任务详情与深链
VITE_AIPI_CHAT_URL	AIPI 对话类接口	否	对应智能体/业务
VITE_FORCE_COLLAPSED_SIDEBAR	强制紧凑侧边栏	否	刷新后导航形态
Logo/欢迎页相关变量	品牌素材 URL	否	明暗主题、登录页

7.5 修改后的生效

1. 备份 runtime-config.js，确认 JavaScript 逗号和引号语法正确。
2. 重启 Web 容器，或至少确认挂载文件已经更新。
3. 浏览器使用 Ctrl+F5 强制刷新；必要时清空站点缓存。
4. 在开发者工具 Network 中检查 runtime-config.js、/deerflow/health、/deerflow/api 请求。
5. 验证普通登录、对话流和至少一个外部业务接口。

7.6 前端常见现象

现象	检查	处理
页面白屏	Console 是否有语法/资源错误	修复 runtime-config.js；恢复备份；强刷
页面能开但所有 API 失败	Network 请求地址	修正 BACKEND/LANGGRAPH_BASE_URL
刷新二级路由 404	Nginx 是否 SPA try_files	恢复随包 Nginx 配置并重建 Web
502 Bad Gateway	backend 状态与日志	先恢复后端，不要反复刷新前端
CORS blocked	前后端 Origin 是否不同	优先改同源；或配置后端 CORS
Mixed Content	HTTPS 页面调用 HTTP	接口改 HTTPS 或统一由 Nginx 代理

8. 技能：目录、上传、发布与维护

技能由宿主机 data/skills 持久化到容器 /app/backend/skills。系统使用 public 和 custom 两类目录，每个技能至少应包含 SKILL.md；还可包含 Python 脚本、references 和 assets。启动脚本只会在技能挂载目录为空时写入默认技能，不会覆盖已有文件。

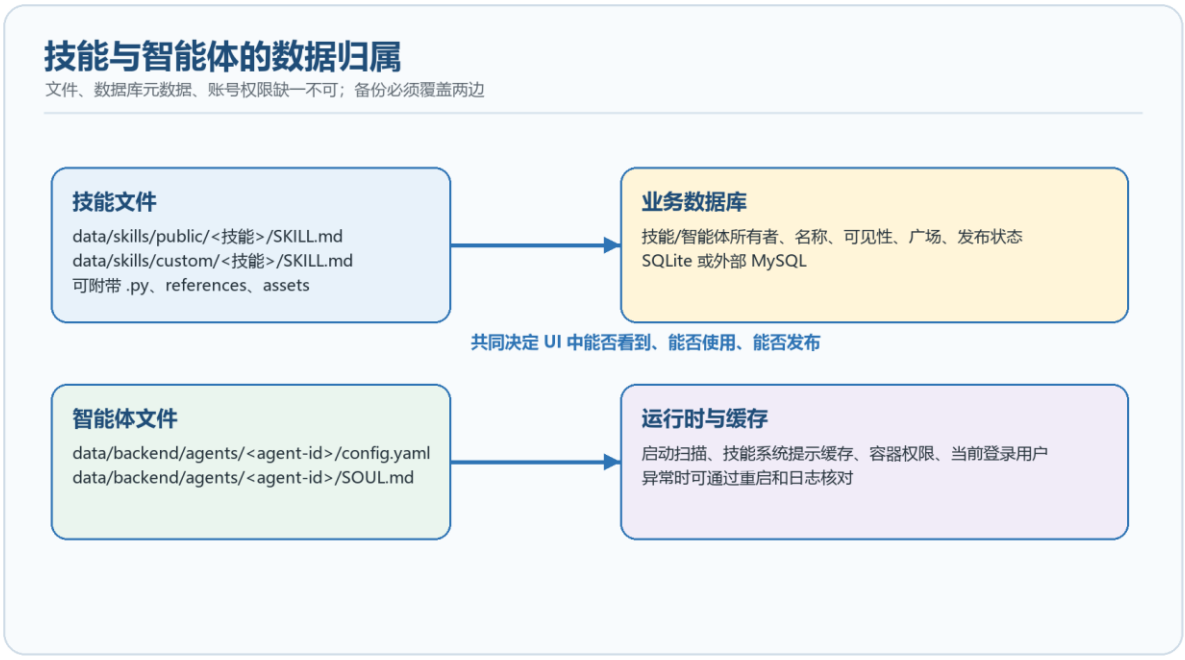


图 8-1 技能与智能体的持久化关系

8.1 标准技能目录

目录结构

```
data/skills/
├── public/
│   └── example-public-skill/
│       ├── SKILL.md
│       ├── scripts/
│       ├── references/
│       └── assets/
└── custom/
    └── example-custom-skill/
        ├── SKILL.md
        └── tool.py
```

8.2 推荐：通过前端上传或创建

1. 使用普通业务用户或管理员登录，进入技能设置/技能管理。
2. 创建或上传技能，填写名称、描述、使用说明和依赖。
3. 在技能详情页完成测试，确认工具地址、Python 依赖和文件权限。
4. 需要共享时点击“发布”，选择目标广场；不共享则保留为自定义技能。

> **为什么推荐 UI** UI 会同时维护技能文件、所有者、可见性、发布状态等数据库元数据。手工复制文件只解决文件存在，不一定能让正确用户在界面看到。

8.3 手工离线导入技能

仅在批量迁移或 UI 不可用时使用。先在测试环境核对技能内容，再复制整个技能目录，保持目录名稳定。

Shell

```
# 例：导入为自定义技能
sudo mkdir -p data/skills/custom/example-skill
sudo cp -a /mnt/offline/example-skill/. data/skills/custom/example-skill/

# 核对宿主机文件
find data/skills/custom/example-skill -maxdepth 3 -type f -print

# 核对容器内挂载
sudo docker compose exec backend sh -lc       'find /app/backend/skills/custom/example-skill -
maxdepth 3 -type f -print'
```

若容器不是 root 运行，需确保运行用户对文件可读、对需要修改的目录可写。完成后重启 backend 以刷新技能扫描与系统提示缓存。

8.4 public 与广场发布的区别

概念	含义	建议权限	是否等于广场可见
public 目录	技能文件类别/系统公共技能	仅管理员直接写入	不一定，仍受业务元数据和页面逻辑影响
custom 目录	用户自定义技能	普通用户维护自己的	可发布到广场
发布广场	业务可见性、目标广场和发布状态	所有者或管理员	是

8.5 技能地址管理

管理员可在技能管理页的“技能地址管理”标签扫描运行中 skills/public 与 skills/custom 下 .md、.py 中出现的 HTTP(S) URL 和 IP:端口，预览后批量替换，并按修改记录回退。适用于 UAT 地址切生产地址。

- 替换前使用预览，核对每个 file:line 片段；不要直接全目录文本替换。
- 应用会生成磁盘备份和 manifest；回退前仍建议做整目录备份。
- 扫描范围只包含运行中技能目录，不会修改 deploy/minimal/skills。
- 接口为管理员权限；大量技能扫描时等待进度完成，不要重复点击。

8.6 技能发布到广场

1. 确认当前登录用户是技能所有者，或使用 admin。
2. 在技能列表/详情点击“发布”。
3. 选择一个发布广场；未选择时可能进入系统默认广场。
4. 确认发布后，使用普通用户进入广场检索并安装/使用。
5. 需要改投时重新发布到另一个广场；一个技能同一时间只属于一个广场。

8.7 技能上传安全检查

- 检查 SKILL.md 和脚本中是否包含真实 API Key、数据库密码、内网敏感地址或个人信息。
- 检查 Python/shell 脚本是否包含删除、提权、任意命令执行或越权网络访问。
- 需要外部依赖时，离线镜像必须已包含包，或提供受控离线 wheel/npm 包。
- 仅允许可信管理员把技能直接放入 public；普通用户走 custom + 广场审批。

9. 智能体：目录、模型、发布与恢复

智能体的主要文件位于 data/backend/agents/<agent-id>/，通常包含 config.yaml 与 SOUL.md；名称、所有者、发布状态、广场等信息在业务数据库中。因此备份或迁移智能体必须同时覆盖 data/backend 和数据库。

9.1 标准目录与职责

目录结构

```
data/backend/agents/<agent-id>/
├── config.yaml    # 模型、工具、参数等结构化配置
└── SOUL.md        # 角色、任务、边界、表达风格等提示词
```

对象	主要内容	缺失影响
config.yaml	模型引用、工具/技能和运行参数	智能体可能无法加载或退化为最小配置
SOUL.md	角色提示词和业务规则	角色行为改变、回答质量下降
数据库记录	所有者、名称、状态、发布信息	界面看不到、权限异常、广场丢失

9.2 创建与维护建议

1. 优先在智能体管理页面创建，系统会生成 ID 和数据库记录。
2. 选择已在后端 models 中存在的模型 name；模型改名时先盘点引用。
3. 配置工具/技能后完成一轮真实问答和错误场景测试。
4. SOUL.md 中不要写真实密钥；外部账号通过环境变量或受控凭据服务注入。
5. 修改生产智能体前导出目录和数据库备份，保留变更说明。

9.3 发布到广场

智能体列表卡片或详情页提供发布入口。所有者或管理员选择目标广场并确认；发布后普通用户可在对应广场发现。重新发布到另一广场表示移动，不会在多个广场重复复制。私有广场状态与公开发布状态应按页面规则保持互斥。

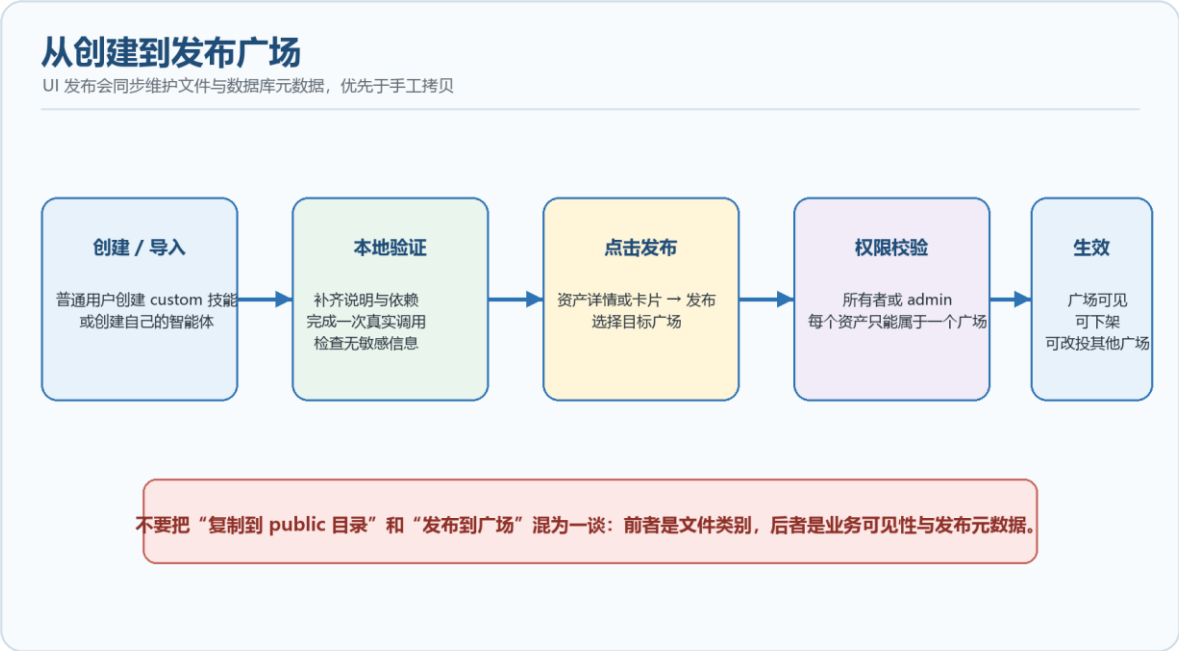


图 9-1 技能/智能体发布流程

9.4 直接发布与私有广场

- “直接发布到广场”仍建议从资产卡片/详情点击发布，这样所有者和发布元数据完整。
- 管理员在外观/系统设置中维护广场列表、默认广场和私有广场密码。
- 若资产已在私有广场，转为公开发布前按页面提示解除私有状态；反向操作同理。
- 发布后要用普通用户验证，不要只看管理员列表。

9.5 智能体模型失效

现象	原因	处理
智能体加载但对话报模型不存在	config.yaml 引用了已删除/改名的模型 name	恢复原 name 或批量更新智能体配置
模型列表有但智能体仍用旧模型	智能体配置固定引用旧值或缓存未刷新	修改智能体并重启 backend
同名智能体权限不同	数据库所有者/作用域不同	用 admin 检查记录，不要仅比较目录名

10. 发布广场与私有广场管理

发布广场是技能和智能体的分发入口。管理员维护广场列表、默认广场、启用状态与私有广场密码；资产所有者或管理员负责发布和下架。

10.1 管理员配置广场

1. 使用 DeerFlow admin 登录。
2. 进入设置中的外观/发布广场配置区域。
3. 新增广场名称和说明，设置启用状态、排序和默认广场。
4. 如启用私有广场，配置独立密码并通过安全渠道分发。
5. 保存后用普通用户检查广场入口、默认选择和访问控制。

10.2 发布规则

规则	解释	运维影响
一项资产只属于一个广场	重新发布到新广场会移动	迁移前通知原广场使用者
默认广场	未明确选择时作为默认目标	删除/停用前先指定新默认
所有者或管理员可发布	防止普通用户操作他人资产	账号变更时先转移所有权或由 admin 处理
私有与公开互斥	防止同一资产同时处于两类可见性	变更后用两类用户复核

10.3 看不到发布按钮

- 当前用户不是资产所有者，也不是 admin。
- 资产来自系统公共目录，页面只允许管理员操作。
- 发布广场全部停用或没有默认广场。
- 前端缓存、权限令牌过期，或后端数据库切换后记录不完整。
- 打开浏览器 Network，检查 publish API 的 401/403/409 响应。

11. 菜单名称、排序、隐藏与路由安全

管理员可通过“设置和更多 → 菜单管理”修改侧边栏节点的显示名称、启停、顺序和父级。系统以静态路由树为基础，仅把数据库中的覆盖项叠加上去；图标和实际 path 不会因为改名称而改变。

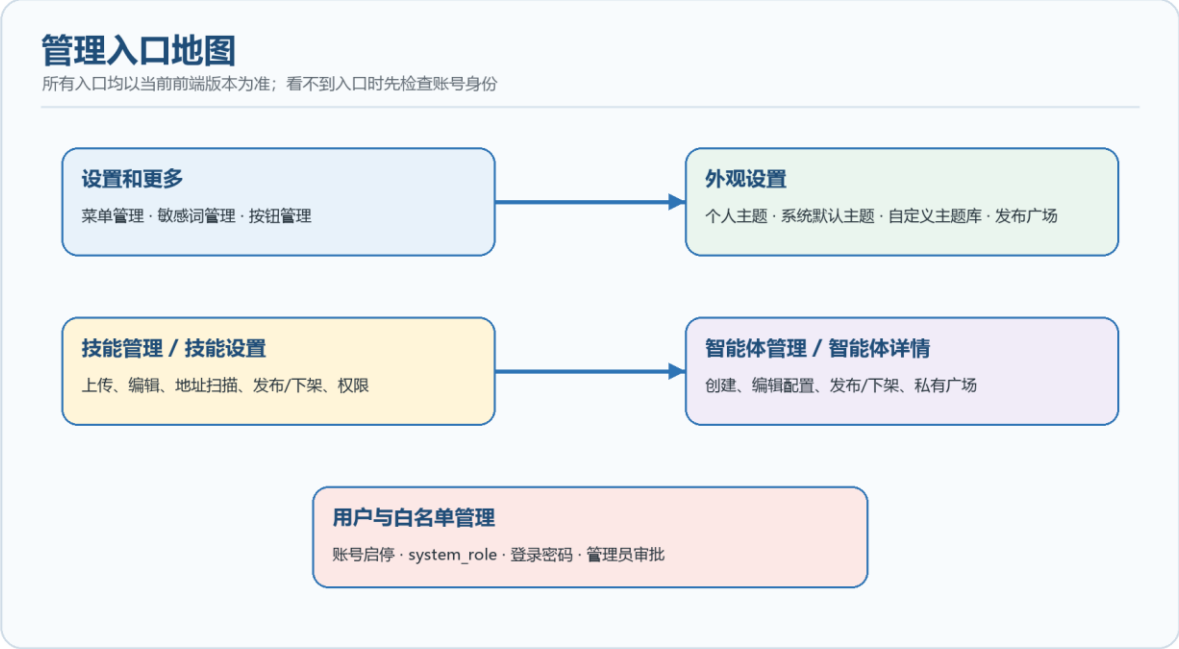


图 11-1 常用管理入口地图

11.1 修改菜单名称

1. 用 admin 登录，打开左下角“设置和更多”。
2. 进入“菜单管理”，找到目标一级/二级/三级菜单。
3. 编辑 label，保存；不要修改真实 path 或源代码 ID。
4. 刷新页面，用普通用户核对侧边栏和顶部导航。
5. 记录原名称，避免业务培训资料与新名称不一致。

11.2 隐藏菜单

在菜单管理中将节点停用。停用父节点会隐藏整个子树；支持上下移动或拖拽调整顺序和父级，但系统严格限制最多三级。动态注入的 iframe 轻应用节点也可管理。

> **重要安全边界** 隐藏菜单只移除导航入口，不保证用户不能通过已知 URL 直接访问，也不替代后端 API 权限。敏感功能必须同时有前端路由守卫和后端角色校验。

11.3 需要真正禁用功能时

1. 先停用菜单，避免用户继续进入。
2. 检查对应页面是否有 admin/lqq/业务角色路由守卫。
3. 检查后端 API 是否校验角色、所有者或账号。
4. 必要时通过功能开关或后端配置关闭 API；不要只依赖 CSS 隐藏。
5. 用普通账号直接输入 URL 和直接调用 API 进行验收。

11.4 菜单异常恢复

问题	原因	处理
全部子菜单消失	父节点被停用或移到错误父级	admin 在菜单管理恢复父节点；必要时查 menu-overrides 数据
菜单名改了但路由未变	这是设计行为	继续使用原 path；不要以显示名推导 URL
轻应用菜单消失	轻应用停用、父级停用或 override 隐藏	同时检查轻应用和菜单覆盖
普通用户看不到，admin 能看到	角色过滤或功能权限	确认预期角色，不要直接改静态路由

12. 主题色、布局和品牌配置

所有用户都可在“设置 → 外观”选择个人主题和布局。管理员还可配置系统默认主题、默认布局和自定义主题库。用户已有个人选择时，个人设置通常优先于系统默认。

12.1 内置主题

主题标识	界面名称	使用建议
light	标准浅色	默认办公环境
dark	深色	暗光环境和大屏
lightblue	浅蓝	政企蓝色体系
lightgreen	浅绿	运营/生态类场景
iceblue	冰蓝	冷色、低饱和界面
lightyellow	浅黄	暖色、阅读型界面

12.2 管理员设置系统默认

1. 使用 admin 登录，进入设置 → 外观。
2. 设置系统默认主题和默认布局；这些主要影响没有个人覆盖的新用户。
3. 在自定义主题库中配置主题名称、主色、启用状态和排序。
4. 保存后使用新普通用户或清除个人覆盖的测试用户验证。
5. 分别检查明暗主题的 Logo、文字对比度、按钮和焦点状态。

12.3 自定义主题色的验收

- 正文文字和背景必须有足够对比度；不要只看主按钮。
- 检查 hover、selected、disabled、focus、error、success 等状态。
- 检查表格、弹窗、下拉菜单、图表、侧边栏和移动窗口宽度。

- 品牌主色不要直接当大面积背景，浅色主题应提供足够浅的层级色。
- 主题调整后至少用 Chrome/Edge 验证，并保留一键恢复内置主题的方法。

12.4 强制紧凑侧边栏

JavaScript

```
// config/runtime-config.js
window.__DEERFLOW_RUNTIME_CONFIG__ = {
  VITE_FORCE_COLLAPSED_SIDEBAR: "true"
};
```

这是部署级开关，适用于外部嵌入或小屏展示。若用户报告无法展开侧边栏，先检查该值。

12.5 iframe 嵌入主题

外部系统可在深链加入 ?embed=1&theme=dark 等参数隐藏全局导航并强制当前标签页主题。该设置使用会话级隔离，不应覆盖普通标签页的个人主题。另有 ?hideJump=1 隐藏快捷跳转栏，?isolate=1 启用本标签页独立登录会话。

13. 敏感词管理与显示层脱敏

敏感词功能维护“原始词 → 替换词”的展示层映射。它会改变菜单/导航等页面显示，但不会修改原始业务数据、路由 path、节点 ID 或数据库中的源文本。因此它不是数据清洗，也不能代替数据库脱敏。

13.1 登录哪个用户

> **使用 lqq** 当前前端入口、路由守卫和页面守卫要求显示用户名转小写后等于 lqq。即使 admin 后端有写权限，非 lqq 管理员通常也看不到或进不了该页面。

13.2 操作步骤

1. 以 lqq 登录，进入左下角“设置和更多 → 敏感词管理”。
2. 新增原始词和替换词，设置启用状态；避免把过短的常用单字作为原始词。
3. 如需恢复系统初始映射，使用“导入种子词”合并缺失项。
4. 保存会整体替换当前配置，提交前导出或截图留档。
5. 刷新页面，检查菜单、导航、标题；再用普通用户验证。

13.3 设计与风险

问题	说明	建议
是否改数据库原文	不会，只影响展示	需要源数据脱敏时另走数据治理流程
是否影响 URL	不会，path/id 保持不变	文档仍使用真实路由
空配置怎么办	加载失败或空时保留前端种子映射	检查 /api/sensitive-words 与日志
替换冲突	多个词重叠可能产生意外显示	从长词到短词测试；避免循环式替换
保存影响范围	全局共享	必须有审批和变更记录

13.4 页面看不到时

- 确认显示用户名是 lqq，而不是仅系统角色为 admin。
- 邮箱登录时确认 @ 前缀；清理旧 Token 后重新登录。
- 确认菜单覆盖没有停用“敏感词管理”节点。
- 检查后端 /api/sensitive-words 的 401/403 和数据库连接。

14. 备份、恢复、升级与回滚

一次可恢复的备份至少包含配置、持久化文件和当前业务数据库。只保存 Docker 镜像不能恢复用户、技能、智能体和全局设置。外部 MySQL 必须单独导出。

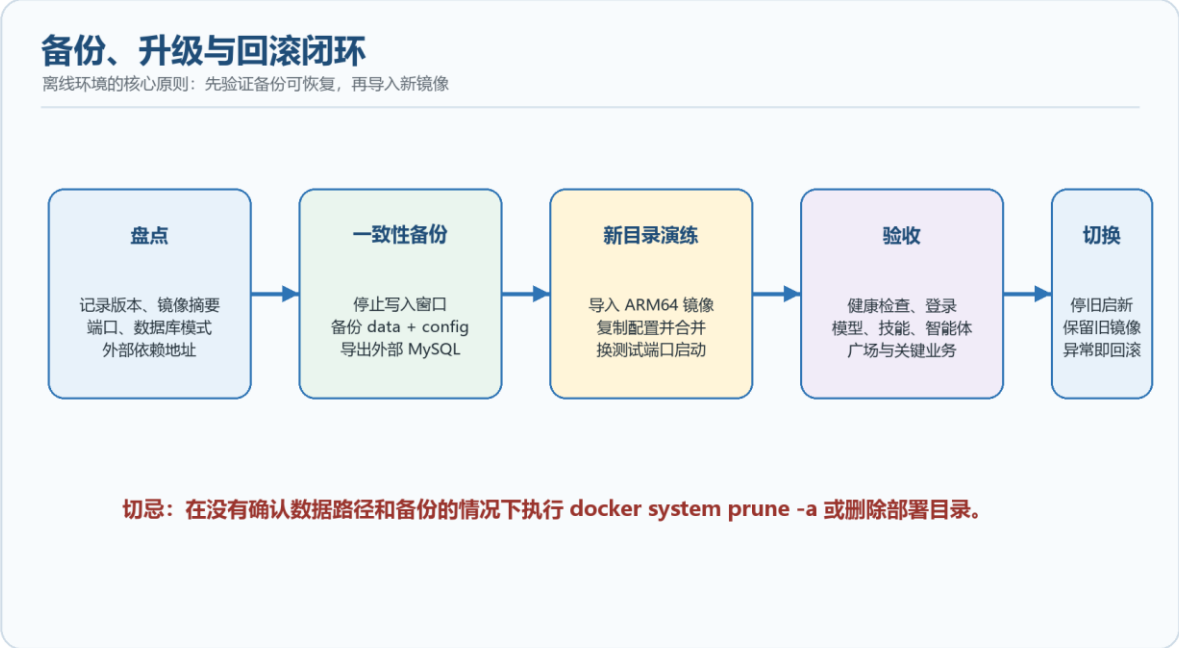


图 14-1 离线升级与回滚闭环

14.1 备份范围

对象	必须	建议频率	恢复用途
config 目录 + .env	是	每次变更后	恢复模型、登录、接口、数据库配置
data/backend	是	每日 + 变更前	SQLite、智能体文件和运行数据
data/skills	是	每日 + 技能变更后	public/custom 技能
data/knowledge	按使用	每日/每周	知识库文件
外部 MySQL dump	使用 MySQL 时	每日 + 变更前	资产元数据、用户、菜单、主题、广场等
镜像 tar + SHA256	是	每个版本	离线重装和回滚

14.2 文件备份示例

Shell

```
cd /opt/deerflow/deerflow-deploy

# 推荐先安排短暂停写窗口，再执行归档
sudo tar -czf /backup/deerflow-files-$(date +%F-%H%M).tar.gz       .env config data

sha256sum /backup/deerflow-files-*.tar.gz
```

14.3 MySQL 备份示例

Shell

```
# 不要把密码直接写在命令行历史中；优先使用受控 defaults-extra-file
mysqldump --single-transaction --routines --triggers       --host=<MYSQL_HOST> --
user=<BACKUP_USER> deerflow       > /backup/deerflow-mysql-$(date +%F-%H%M).sql
```

14.4 恢复原则

1. 确认备份版本、摘要、数据库模式和目标目录；禁止直接覆盖未知现网。
2. 停止写入或停止 backend，先把现网再做一次应急备份。
3. 在新目录恢复 config 和 data，检查文件所有者与权限。
4. 外部 MySQL 恢复到新库或确认的空库，避免覆盖错误库。
5. 用临时 APP_PORT 启动，完成登录、模型、技能、智能体、广场和对话验收。
6. 切换正式端口；保留旧目录、旧库和旧镜像直到观察期结束。

14.5 离线升级

- 核对离线包架构必须为 linux/arm64；SHA256 与交付清单一致。
- 不要直接覆盖旧目录。解压到带版本号的新目录，复制 data 和配置后逐项合并。
- 先用临时端口启动新版本，数据库升级必须有回退方案。
- 升级后测试管理员、lqq、普通用户三类账号。
- 观察期内保留旧镜像，禁止执行会清理回滚镜像的 prune。

> **高风险命令** docker system prune -a、删除部署目录、覆盖数据库和递归修改所有者均可能造成不可逆损失。执行前必须明确目标、备份已验证并经过审批。

14.6 备份校验与恢复演练

备份任务显示“成功”不代表一定可恢复。至少每季度在隔离目录或隔离服务器做一次完整恢复演练，并记录恢复用时和缺失项。

检查项	检查方法	合格标准
归档完整性	tar -tzf 列出 config、data 关键目录	无读取错误，路径与部署台账一致
文件摘要	重新计算 SHA256 并与备份台账比对	摘要完全一致
SQLite	在副本上执行完整性检查并启动测试环境	检查通过，用户与资产数量正确
MySQL	恢复到隔离库，核对表、行数和关键业务	无迁移错误，三类账号验收通过
应用验证	临时 APP_PORT 启动并执行上线验收清单	健康、登录、模型、资产、管理功能通过
回退时长	记录从故障确认到旧版本恢复的总耗时	满足本单位 RTO/RPO 要求

15. 监控、巡检与日志留存

15.1 每日巡检

- docker compose ps：backend、web 均为正常状态，无反复重启。
- /deerflow/health 返回成功；前端首页可打开。
- 磁盘剩余空间、inode、内存和负载无异常。
- backend/web 日志无持续 401、429、500、数据库断连或权限错误。
- 模型最短问答可用；关键业务接口按单位要求抽测。
- 前一日文件备份和数据库备份成功，摘要记录完整。

15.2 每周巡检

- 检查离线镜像、部署目录和文档版本是否对应。
- 统计模型调用错误率、超时和 429，评估并发与额度。
- 检查技能地址是否指向过期 UAT/测试环境。
- 抽查发布广场、私有广场、菜单、主题和敏感词是否符合审批。
- 在隔离环境随机做一次备份恢复演练。
- 清理日志前先确认保留策略；不要删除仍在被容器写入的文件。

15.3 建议监控指标

层级	指标	建议告警
主机	磁盘、inode、内存、负载、时间同步	磁盘 <20% 预警，<10% 紧急
容器	Up、重启次数、CPU/内存	反复重启或持续高占用
HTTP	首页、/health、5xx、延迟	健康失败、连续 5xx
模型	401/403/404/429、超时、首 token 延迟	权限错误立即，429 持续告警
数据库	连接数、慢查询、空间、备份	连接耗尽、备份失败

层级	指标	建议告警
业务资产	技能/智能体数量、发布变更、管理员操作	数量突降、未经审批的全局变更

15.4 日志采集包

提交问题时应包含：发生时间（含时区）、用户和角色、访问 URL、操作步骤、浏览器 Network/Console、backend/web 最近日志、compose ps、健康检查、版本号与配置变更。任何日志对外发送前都要脱敏。

16. 常见报错与解决办法

16.1 容器和主机

报错/现象	根因	解决
exec format error	镜像架构不匹配	确认 uname -m；ARM 服务器使用 linux/arm64 镜像
address already in use	APP_PORT 被占用	ss -lntp 检查；改 .env 或停冲突服务
no space left on device	磁盘或 inode 满	先定位大目录和日志；扩容或受控清理；不要盲目 prune
容器不断 Restarting	配置解析、数据库、权限或启动命令失败	docker compose logs --tail=500 backend
permission denied	挂载目录所有者/模式不匹配	对照容器 UID/GID 修复目标目录，不要全盘 chmod 777
镜像导入失败	tar 损坏或磁盘不足	核对 SHA256、剩余空间、docker load 日志

16.2 前端和 Nginx

报错/现象	根因	解决
502 Bad Gateway	backend 未启动或 Nginx 无法访问 backend:8001	查 backend 日志和 compose 网络；先恢复后端
404 刷新二级路由	SPA fallback 丢失	Nginx location / 使用 try_files ... /index.html
静态资源 404	错误 base 路径、缓存或镜像不完整	查 Network 路径、runtime-config 和 Web 镜像
页面白屏	JavaScript 语法、旧缓存、运行时配置异常	看 Console；恢复 runtime-config；Ctrl+F5
请求发到旧地址	浏览器缓存或改错文件	确认 Network 中 runtime-config.js；重启 Web 并强刷

16.3 登录和权限

报错/现象	根因	解决
401 Unauthorized	Token 过期、JWT 密钥变化或存储隔离	重新登录；核对 AUTH_JWT_SECRET 是否稳定
403 Forbidden	角色/所有者不足	核对 system_role、资产所有者、lqq 条件
重启后全员掉线	JWT 密钥被改或未持久化	恢复固定密钥；全员重新登录
admin 看不到敏感词入口	前端要求用户名 lqq	使用 lqq；不要仅提升 admin
iframe 与正常页账号混淆	未使用 embed/isolate 会话隔离	核对 URL 参数与 sessionStorage 策略

16.4 模型和流式对话

报错/现象	根因	解决
模型 401/403	Key 或权限	核对 backend.env、模型授权、IP 白名单
模型 404	base_url/model ID	用提供方接口清单核对，注意 /v1
模型 429	并发/限流/余额	降并发、等待、申请额度；避免重试风暴
流式输出中断	代理缓冲、超时、客户端断网	查 Nginx SSE 配置、后端日志和 Network
回答截断	max_tokens 或上游输出上限	逐步调大，查 finish_reason；不要超过提供方限制
工具调用循环	模型能力、提示词或工具返回异常	查看线程日志，暂时禁用问题工具并修复输入输出

16.5 数据库和资产

报错/现象	根因	解决
MySQL Access denied	账号、密码或 host	核对 URL 编码和授权

报错/现象	根因	解决
启动迁移报错	版本跨度、权限或脏迁移	停止反复启动，备份后按迁移日志处理
技能存在但 UI 不显示	数据库所有者/作用域或扫描失败	查文件、记录、权限和 backend 日志
智能体目录存在但列表为空	数据库切换或元数据缺失	恢复同版本数据库或按管理流程重建记录
发布后广场无资产	发布目标、过滤、私有状态或缓存	用普通用户查目标广场和 publish API

17. 技能或智能体丢失专项排查

“丢失”不一定是磁盘文件被删。账号变化、数据库切换、发布过滤、错误 Compose 目录、空挂载或权限失败都可能表现为界面看不到。按固定顺序排查，避免先复制文件造成数据进一步分叉。

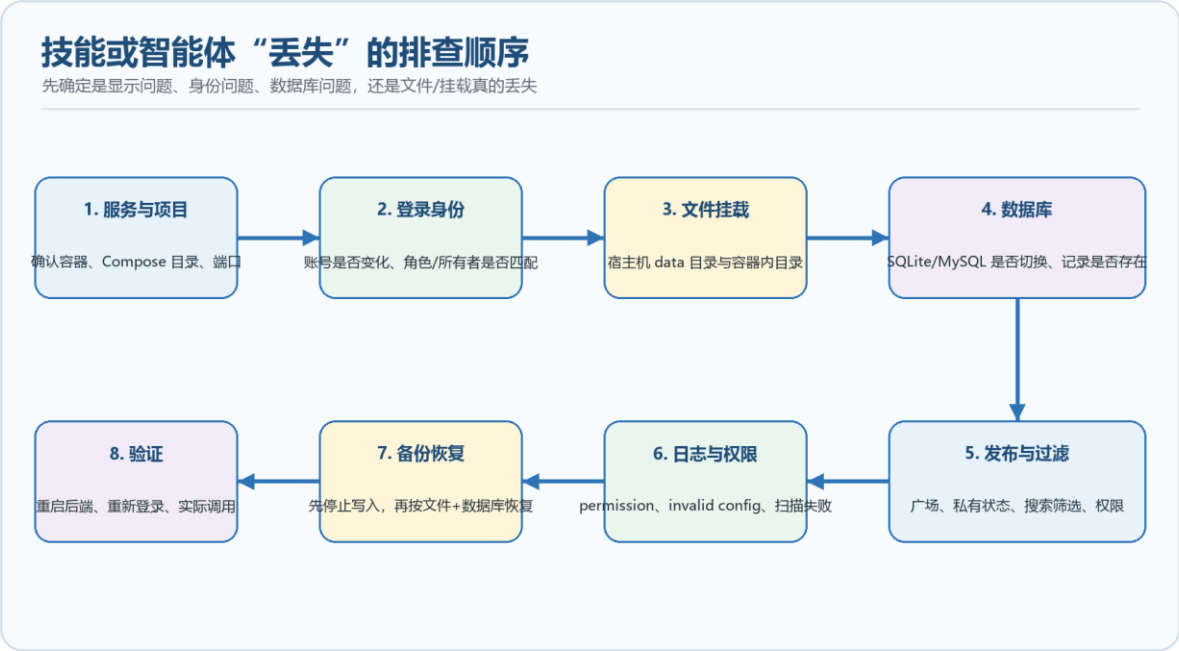


图 17-1 资产丢失排查流程

17.1 第一步：确认当前服务和部署目录

Shell

```
pwd
sudo docker compose ps
sudo docker compose config --services
sudo docker compose images
```

重点排除在旧部署目录执行命令，或 1Panel 启动了另一个同名 Compose 项目。记录容器名称、镜像 ID 和端口。

17.2 第二步：确认登录身份

- 退出重新登录，确认用户名、system_role 和是否使用 iframe/isolate 独立会话。
- 确认资产原所有者是谁；普通用户不能看到他人私有资产。
- 用 admin 查看全局资产；再用原所有者确认。

17.3 第三步：核对宿主机与容器内文件

Shell

```
# 宿主机
find data/skills -maxdepth 3 -type f -print | sort
find data/backend/agents -maxdepth 3 -type f -print | sort

# 容器内
sudo docker compose exec backend sh -lc       'find /app/backend/skills -maxdepth 3 -type f -
print | sort'
sudo docker compose exec backend sh -lc       'find /app/backend/.deer-flow/agents -maxdepth 3
-type f -print | sort'
```

如果宿主机有、容器内没有，检查 docker-compose.yml 的 volume 映射和当前工作目录；如果两边都没有，先停止写入并查备份，不要创建同名空资产覆盖恢复线索。

17.4 第四步：确认数据库没有切换

- 检查 config.yaml 的 database.backend 是 sqlite 还是 mysql。
- 检查 backend.env 的 MYSQL_DATABASE_URL 指向哪个主机和库。
- 核对变更前后资产数量；SQLite 与 MySQL 不会自动保持同步。
- 如果切回旧数据库后资产恢复，先冻结写入，制定合并方案，避免两套库继续分叉。

17.5 第五步：检查过滤、广场和权限

- 清除搜索词和分类筛选；查看“我的”“公共”“已发布”“私有”等不同视图。
- 确认资产是否改投另一广场、被下架、进入私有广场或所有权变化。
- 检查 menu-overrides 是否隐藏了技能/智能体入口。

17.6 第六步：日志和权限

Shell

```
sudo docker compose logs --tail=1000 backend       | grep -Ei
'skill|agent|permission|database|mysql|sqlite|invalid|error|traceback'
```

重点关注：SKILL.md 格式不合法、config.yaml 解析失败、目录 permission denied、数据库连接失败、扫描缓存刷新失败和 401/403。

17.7 恢复决策

文件	数据库	建议
存在	存在	修复权限、挂载、账号或过滤；重启 backend 验证
存在	缺失	从同版本数据库备份恢复元数据；不要直接伪造所有者
缺失	存在	从文件备份恢复；数据库可能只能提供最小信息
缺失	缺失	停止写入，恢复完整备份；保全日志和磁盘证据
两套库各有新数据	分叉	冻结写入，导出比对，编写一次性合并迁移并审计

> **不要做什么** 不要执行 docker system prune -a，不要删除 data 重新启动，不要把空目录覆盖到挂载点，不要用 chmod -R 777 掩盖所有权问题。

18. 安全加固与密钥管理

18.1 密钥清单

密钥/凭据	保存位置	前端可见	轮换建议
AUTH_JWT_SECRET	config/backend.env	绝对不可	事件触发或年度；轮换会使会话失效
模型 API Key	config/backend.env	绝对不可	按提供方策略，泄露立即轮换
MySQL URL/密码	config/backend.env 或安全挂载	绝对不可	90~180 天或按制度
私有广场密码	后台全局设置/数据库	受登录用户可见范围	人员变化或定期
runtime-config 地址	config/runtime-config.js	是	只能放非秘密地址和开关

18.2 最低安全要求

- 服务器 SSH 禁止长期使用共享 root 密码，启用个人密钥、最小 sudo 和登录审计。
- 1Panel 使用独立安全入口、强密码、双因素认证和访问白名单；及时安装受控补丁。
- 只开放 APP_PORT、SSH 管理端口和必要数据库访问；后端 8001 默认不对外映射。
- config/backend.env 权限建议 600，部署目录限制给运维组；备份同样加密和访问控制。
- 审计 config.yaml，包括注释中的历史 API Key；发现真实 Key 即轮换并删除，不因注释状态降低风险。
- 技能和智能体提示词中禁止嵌入密钥；上传脚本按代码审查流程执行。
- 内网不等于可信。模型、数据库和业务接口仍使用 TLS、账号权限和日志审计。

18.3 日志脱敏

- 对外发送日志前删除 Authorization、Cookie、Token、API Key、数据库 URL、手机号和身份证号。
- 保留时间戳、请求路径、状态码、trace ID 和错误栈，避免过度脱敏导致无法定位。

- 不要在群聊粘贴完整 backend.env；必要时只报告变量是否已设置和哈希/末四位。

18.4 离线介质

- 导入前扫描移动介质；镜像和文档核对 SHA256。
- 记录介质编号、导入人、时间和文件清单。
- 导入完成后按单位制度清除临时介质和服务器临时目录。

19. 标准运维流程与交接清单

19.1 上线验收

类别	验收项	结果
基础	CPU 架构为 ARM64；镜像 SHA256 与交付清单一致	□
服务	web/backend 正常；健康接口成功；无重启循环	□
网络	APP_PORT 可访问；8001 未无意暴露；外部依赖可达	□
登录	admin、lqq、普通用户均可登录且权限符合预期	□
模型	短问答、长文本、工具调用和视觉能力按需通过	□
资产	技能、智能体数量一致；所有者、广场和私有状态正确	□
管理	菜单、主题、敏感词、发布广场保存并对普通用户生效	□
备份	文件和数据库备份成功，并完成至少一次恢复抽检	□
文档	端口、地址、版本、账号责任人和回退步骤已登记	□

19.2 变更单最少信息

- 变更目标、影响范围、窗口、负责人和审批单号。
- 修改前后值（密钥只记录变量名，不记录明文）。
- 备份位置、摘要、恢复验证结果。
- 执行命令、服务重启范围、健康和业务验收结果。
- 回退触发条件、回退命令、回退负责人。

19.3 离职/交接

- 回收 Linux/1Panel、DeerFlow、MySQL、模型平台和外部系统账号。
- 轮换共享密钥、私有广场密码和应急账号密码。
- 核对定时备份、监控、证书和离线介质责任人。

- 移交最近一次升级记录、已知问题和未完成变更。

19.4 运维责任矩阵

事项	Linux/1Panel 运维	DeerFlow admin	模型/数据库管理员
容器、端口、磁盘	主责	知会	知会
菜单、主题、广场、用户	支持	主责	知会
模型参数与密钥	执行部署	业务验收	主责并授权
数据库迁移与恢复	容器与文件支持	业务验收	主责
技能/智能体丢失	核对挂载和备份	核对所有者和元数据	核对数据库
安全事件	隔离主机与日志保全	冻结应用账号/资产	轮换凭据并保全审计

19.5 交接会必须确认

- 现场演示一次后端环境变量变更后的强制重建、实时日志和一键诊断。
- 现场演示 admin、lqq、普通用户三类账号的边界。
- 现场定位技能、智能体文件目录和当前数据库，并抽查最近备份。
- 明确故障升级联系人、响应时间、离线介质和回退决策人。

附录 A：命令速查

A.1 服务

Shell

```
sudo bash install.sh                      # 首次安装/加载镜像并启动
sudo bash restart.sh                      # 重启全部
sudo bash stop.sh                         # 停止全部
sudo docker compose restart backend       # 日常重启后端/挂载配置变更
sudo docker compose up -d --force-recreate backend  # backend.env 变更
sudo docker compose restart web           # 只重启前端
sudo docker compose ps                    # 状态
```

A.2 日志与健康

Shell

```
sudo bash logs.sh                         # 全部实时日志
sudo bash logs.sh backend                 # 后端实时日志
sudo bash logs.sh web                     # Web/Nginx 实时日志
sudo env TAIL=1000 bash diagnose.sh       # 一键诊断
curl -fsS http://127.0.0.1:8090/deerflow/health
```

A.3 主机资源

Shell

```
uname -m                                  # 架构，应为 aarch64/arm64
df -h                                     # 磁盘空间
df -i                                     # inode
free -h                                   # 内存
ss -lntp                                  # 端口监听
sudo docker stats --no-stream             # 容器资源
```

A.4 文件与挂载

Shell

```
sudo docker compose config                # 展开后的 Compose 配置
sudo docker compose exec backend sh -lc 'id; pwd; mount'
sudo docker compose exec backend sh -lc 'ls -la /app/backend/skills'
sudo docker compose exec backend sh -lc 'ls -la /app/backend/.deer-flow/agents'
```

附录 B：安全配置模板

B.1 部署 .env

ENV

```
COMPOSE_PROJECT_NAME=deerflow-1panel
# 原样复制离线包内 .env 的标签值
DEERFLOW_TAG=<PACKAGE_TAG>
APP_PORT=8090
TZ=Asia/Shanghai

# 不要只凭 tag 名称判断 CPU 架构；用 docker image inspect 核对 Architecture
```

B.2 backend.env

ENV

```
AUTH_JWT_SECRET=<强随机值>
INTRANET_MODEL_API_KEY=<模型密钥>
MYSQL_DATABASE_URL=mysql+asyncmy://<USER>:<URL_ENCODED_PASSWORD>@<HOST>:3306/<DB>?charset=utf8m
b4

GATEWAY_ENABLE_DOCS=false
GATEWAY_WORKERS=1
```

B.3 runtime-config.js

JavaScript

```
window.__DEERFLOW_RUNTIME_CONFIG__ = {
  VITE_BACKEND_BASE_URL: "/deerflow",
  VITE_LANGGRAPH_BASE_URL: "/deerflow/api",
  VITE_API_BASE_URL: "<可选业务 API 地址>",
  VITE_AIPI_CONSUMER_URL: "<可选任务业务地址>",
  VITE_FORCE_COLLAPSED_SIDEBAR: "false"
};
```

B.4 模型条目

YAML

```yaml
models:
  - name: intranet-model
    display_name: 内网模型
    use: langchain_openai:ChatOpenAI
    model: <provider-model-id>
    api_key: $INTRANET_MODEL_API_KEY
    base_url: http://<MODEL_HOST>:<PORT>/v1
    request_timeout: 600.0
    max_retries: 2
    max_tokens: 8192
    temperature: 0.7
    supports_vision: false
    supports_thinking: false
```

附录 C：问题升级模板

向研发或供应商提交问题时，复制以下模板填写。密钥和个人信息必须脱敏。

故障单

```
【问题标题】
【发生时间】YYYY-MM-DD HH:mm:ss（Asia/Shanghai）
【环境/版本】生产/测试；离线包版本；镜像 ID；主机架构
【影响范围】全部用户/指定用户；登录/对话/技能/智能体/广场等
【用户身份】admin / lqq / 普通用户（不要提供密码）
【复现步骤】1... 2... 3...
【期望结果】
【实际结果】
【HTTP 状态码 / trace ID】
【最近变更】配置、模型、数据库、镜像、菜单、主题等
【容器状态】docker compose ps
【健康检查】/deerflow/health 结果
【日志附件】backend/web 最近 1000 行（已脱敏）
【浏览器附件】Console + Network（已脱敏）
【已尝试操作】
【是否可回退】是/否；回退点
```

C.1 现场信息保全

- 故障期间先记录时间、容器状态和日志，再重启；重启会丢失部分现场。
- 涉及资产丢失时不要新建同名资产，不要修改数据库，不要清理 Docker。
- 涉及安全事件时立即隔离、轮换凭据并按本单位安全事件流程上报。

附录 D：核心事实速记

主题	结论
前端和后端端口	容器内不同：Web 80、Backend 8001；对用户可由 Nginx 统一为 APP_PORT
前端改端口	部署 .env 的 APP_PORT，重建/重启 Web，放行防火墙
前端改接口	config/runtime-config.js；推荐 /deerflow 和 /deerflow/api
后端模型	config/config.yaml；密钥在 config/backend.env
max_tokens	模型单次输出上限，不是上下文窗口
技能位置	data/skills/public 与 data/skills/custom
智能体位置	data/backend/agents/<agent-id>；还依赖数据库元数据
发布广场	资产所有者或 admin 在 UI 发布；每项资产同一时间一个广场
敏感词用户	正常前端管理使用显示用户名为 lqq 的账号
菜单隐藏	只隐藏导航，不等于禁止直接路由和 API
JWT 密钥	AUTH_JWT_SECRET；必须持久化；改变会导致全员重新登录
数据库切换	SQLite ↔ MySQL 不自动迁移；必须备份、迁移、验收
完整备份	config + data + 外部 MySQL + 镜像及 SHA256
资产丢失	先查账号、挂载、数据库和过滤，再决定恢复；不要先 prune/删除

> **手册维护** 每次镜像升级、路由调整、权限变化或数据库迁移后，由应用管理员和 Linux 运维共同复核本手册并更新版本记录。

运维交付确认

以下信息应由接收方在正式上线前补充，并存放在受控运维台账中。不要在本 Word 文档中写明文密码。

项目	填写内容
生产访问地址	
部署目录	
APP_PORT	
镜像版本 / SHA256	
数据库模式与库名	
Linux / 1Panel 责任人	
DeerFlow admin 责任人	
lqq 账号责任人	
备份位置与保留期	
监控告警联系人	
最近一次恢复演练日期	

交付人签字：_____　接收人签字：_____

日期：_____年___月___日